AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions and listings of claims in the application:

LISTING OF CLAIMS:

1. (Currently Amended) A method for operating a device driver, comprising:

providing a device driver comprising an encrypted program code portion of a main process thereof;

decrypting the encrypted program code portion in an initialization process of said device driver, wherein the decrypting is performed by said device driver and the encrypted program code portion to be decrypted is in said device driver's own program;

executing the decrypted program code portion; and

re-encrypting the executed decrypted program code portion in an end process of the device driver, in which said device driver is released, wherein the re-encrypting is performed by the device driver.

2. (Currently Amended) A method for operating a device driver, comprising:

providing a device driver comprising an encrypted program code portion of a main process thereof;

initializing said device driver;

decrypting the encrypted program code portion after the device driver is initialized, wherein the decrypting is performed by said device driver and the encrypted program code portion to be decrypted is in said device driver's own program;

executing the decrypted program code portion;

re-encrypting the executed decrypted program code portion, wherein the re-encrypting is performed by the device driver; and

releasing said device driver in an end process of the device driver, after the re-encrypting of the executed decrypted program code portion.

3. (Currently Amended) A method for operating a device driver, comprising:

providing a device driver comprising an encrypted program code portion of a main process thereof, wherein the encrypted program code portion has been encrypted a first time with a first encryption key and then encrypted a second time with a second encryption key;

primarily decrypting the encrypted program code portion with a first decryption key in an initialization process of the device driver, wherein the primarily decrypting is performed by said device driver and the encrypted program code portion to be primarily decrypted is in said device driver's own program;

secondarily decrypting the decrypted program code portion with a second decryption key after the initialization process is completed, wherein the secondarily decrypting is performed by said device driver;

executing the secondarily decrypted program code portion;

primarily re-encrypting the secondarily decrypted program code portion with the second encryption key, wherein the primarily re-encrypting is performed by said device driver; and

secondarily re-encrypting the re-~~enrypted~~encrypted program code portion with the first encryption key, wherein the secondarily re-encrypting is performed by said device driver before said device driver is released.

4. (Previously Presented) The method as claimed in claim 1, further comprising extracting a numeric value from an application; and creating a key, corresponding to the numeric value, for decrypting and re-encrypting the program code portion in said decrypting and re-encrypting of the program code portion.

5. (Previously Presented) The method as claimed in claim 2, further comprising extracting a numeric value from an application; and creating a key, corresponding to the numeric value, for decrypting and re-encrypting the program code portion in said decrypting and re-encrypting of the program code portion.

6. (Previously Presented) The method as claimed in claim 1, further comprising performing an authentication between an application and said device driver.

7. (Previously Presented) The method as claimed in claim 2, further comprising performing an authentication between an application and said device driver.

8. (Previously Presented) The method as claimed in claim 1, further comprising:

providing an application, which requests the device driver;

utilizing the application to detect whether or not the program code portion of said device driver has been forged before supplying output data to said device driver, and when the program code portion of said device driver has been forged, the application stops outputting the output data to hardware, and

utilizing the device driver to detect whether or not a program code portion of the application has been forged before supplying input data to the application, and when the program code portion of the application has been forged, said device driver stops outputting the input data to the application.

9. (Previously Presented) The method as claimed in claim 2, further comprising:

providing an application, which requests the device driver;

utilizing the application to detect whether or not the program code portion of said device driver has been forged before supplying output data to said device driver, and when the program code portion of said device driver has been forged, the application stops outputting the output data to hardware, and

utilizing the device driver to detect whether or not a program code portion of the application has been forged before supplying input data to the application, and when the program code portion of the application has been forged, said device driver stops outputting the input data to the application.

10. (Previously Presented) The method as claimed in claim 8,

wherein said device driver does not decrypt encrypted data of the application, and

wherein only when the program code portion of said device driver has not been forged, the application decrypts the encrypted data and provides the decrypted data as the output data to said device driver.

11. (Previously Presented) The method as claimed in claim 9,

wherein said device driver does not decrypt encrypted data of the application, and

wherein only when the program code portion of said device driver has not been forged, the application decrypts the encrypted data and provides the decrypted data as the output data to said device driver.

12. (Previously Presented) The method as claimed in claim 1, wherein the device driver communicates between an application arranged at a user level, and hardware arranged at a privilege level.

13. (Previously Presented) The method as claimed in claim 2, wherein the device driver communicates between an application arranged at a user level, and hardware arranged at a privilege level.

14. (Previously Presented) The method as claimed in claim 3, wherein the device driver communicates between an application arranged at a user level, and hardware arranged at a privilege level.